a2

1. (original) A method for generating a tamper resistant version of a software program including a stream of data blocks, comprising:

undertaking a predetermined number of iterations of forward plain text chaining of the blocks followed by backward plain text chaining of the blocks.

2. (original) The method of claim 1, further comprising XORing a first block with an adjacent block to render a chained block.

3. (original) The method of Claim 2, further comprising scrambling chained blocks using a cipher.

4. (original) The method of Claim 3, comprising scrambling a chained block using at least one but not all rounds of the cipher to render a scrambled block before chaining the chained block to another block.

5. (original) The method of Claim 4, comprising descrambling the chained block using only a single round of the cipher to render a result and then XORing the result with an adjacent block.

6. (original) A computer program device, comprising:

a computer program storage device including a program of instructions usable by an encryption computer, comprising:

a2

logic means for chaining a data block to a plain text version of an adjacent block in the stream to render a chained block;

logic means for scrambling the chained block using a first round of a cipher to render a scrambled block; and

logic means for iterating the means for scrambling and chaining using subsequent rounds of the cipher.

7. (original) The computer program device of Claim 6, wherein the means for iterating iterates forward and backward through the stream, using successive rounds of the cipher.

8. (currently amended) A computer system for encrypting a stream of data blocks, comprising a processor programmed to execute method acts including:

(a) receiving a sequence of N blocks;

(b) initializing a previous block variable B;

(c) for i=1 to N, executing a DO loop comprising:

(c)(1) XORing an ith block with B to render a modified ith block;

(c)(2) setting B equal to the modified ith block;

(c)(3) scrambling the modified ith block using at least one round of a cipher;

(c)(4) incrementing "i" by unity and returning to act ([b]c)(1);

(d) initializing a previous block variable B;

(e) for i=N to 1, executing a DO loop comprising:

Q2

(e)(1) XORing an ith block with B, yielding a modified ith block;

(e)(2) setting B to the modified ith block;

(e)(3) scrambling the modified ith block using at least one next round of a cipher;

(e)(4) decrementing "i" by unity and returning to act ([b]c)(1); and

(f) determining whether a predetermined number of iterations have been executed, and if not, returning to act (b) using a next round of the cipher, otherwise outputting an encrypted stream of data blocks.

9. (original) The computer system of Claim 8, wherein the stream of data blocks is established by a computer program.

10. (original) The computer system of Claim 9, wherein a respective round of the cipher is used for each iteration.

11. (original) A method for generating a tamper resistant version of a software program including a stream of data blocks, comprising:

providing a cipher defining rounds;

iterating through the rounds of the cipher by iterating through respective outer loops of forward plain text chaining followed by backward plain text chaining; and

during each forward portion of an outer loop, applying a respective round of the cipher to each block, and during each backward portion of an outer loop, applying a respective round of the cipher to each block.

12. (currently amended) The method of Claim 11, further comprising:

(a) receiving a sequence of N blocks;

(b) initializing a previous block variable B;

(c) for i=1 to N, executing a DO loop comprising:

(c)(1) XORing an ith block with B to render a modified ith block;

(c)(2) setting B equal to the modified ith block;

(c)(3) scrambling the modified ith block using at least one round of a cipher;

(c)(4) incrementing "i" by unity and returning to act ([b]c)(1);

(d) initializing a previous block variable B;

(e) for i=N to 1, executing a DO loop comprising:

(e)(1) XORing an ith block with B, yielding a modified ith block;

(e)(2) setting B to the modified ith block;

(c)(3) scrambling the modified ith block using at least one next round of a cipher;

(e)(4) decrementing "i" by unity and returning to act ([b]c)(1); and

(f) determining whether a predetermined number of iterations have been executed, and if not, returning to act (b) using a next round of the cipher, otherwise outputting an encrypted stream of data blocks.

13. (original) A method for generating a tamper resistant version of a software program including a stream of data blocks, comprising:

scrambling a block using one and only one round of a cipher; then

chaining the block to another block to render a chained block; then

scrambling the chained block using one and only round of the cipher.

14. (currently amended) The method of Claim 13, further comprising:

(a) receiving a sequence of N blocks;

(b) initializing a previous block variable B;

(c) for i=1 to N, executing a DO loop comprising:

(c)(1) XORing an ith block with B to render a modified ith block;

(c)(2) setting B equal to the modified ith block;

(c)(3) scrambling the modified ith block using at least one round of a cipher;

(c)(4) incrementing "i" by unity and returning to act ([b]c)(1);

(d) initializing a previous block variable B;

(e) for i=N to 1, executing a DO loop comprising:

(e)(1) XORing an ith block with B, yielding a modified ith block;

(e)(2) setting B to the modified ith block;

(e)(3) scrambling the modified ith block using at least one next round of a cipher;

(e)(4) decrementing "i" by unity and returning to act ([b]c)(1); and

(f) determining whether a predetermined number of iterations have been executed, and if not, returning to act (b) using a next round of the cipher, otherwise outputting an encrypted stream of data blocks.

15. (original) A computer system for decrypting a stream of data blocks, comprising a processor programmed to execute method acts including:

(a) receiving a sequence of N blocks;

(b) for i= N to 1, executing a DO loop comprising:

(b)(1) reverse XORing an i^{th} block with a $block_{i-1}$;

(b)(2) unscrambling the i^{th} block using a round of a cipher to render an unscrambled block;

(b)(3) determining whether a $block_{i-1}$ exists, and if not, proceeding to act (c), otherwise;

(b)(4) decrementing "i" by unity and returning to act (b)(1);

(c) for i= 1 to N, executing a DO loop comprising:

(c)(1) reverse XORing an i^{th} block with a $block_{i+1}$;

(c)(2) unscrambling the i^{th} block using a single round of a cipher to render an unscrambled block;

(c)(3) determining whether a $block_{i+1}$ exists, and if not, proceeding to act (d), otherwise;

(c)(4) incrementing "i" by unity and returning to act (c)(1);

(d) determining whether a predetermined number of iterations have been executed, and if not, returning to act (b) using a next round of the cipher, otherwise outputting a decrypted stream of data blocks.

16. (original) The computer system of Claim 15, wherein the stream of data blocks is established by a computer program.

17. (original) The computer system of Claim 16, wherein a respective round of the cipher is used for each iteration.